

Please type a plus sign (+) inside this box → ☐

PTO/SB/05 (4/98)
Approved for use through 09/30/2000. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 C.F.R. § 1.53(b))

Attorney Docket No. 2207/8754

First Inventor or Application Identifier Ronny RONEN et al

Title CONFINING AN APPLICATION TO A 32-BIT ADDRESS SPACE SUBSET IN A 64-BIT PROCESSOR

Express Mail Label No.

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

1. ☒ * Fee Transmittal Form (e.g., PTO/SB/17)
(Submit an original and a duplicate for fee processing)
2. ☒ Specification [Total Pages 15]
(preferred arrangement set forth below)
 - Descriptive title of the Invention
 - Cross References to Related Applications
 - Statement Regarding Fed sponsored R & D
 - Reference to Microfiche Appendix
 - Background of the Invention
 - Brief Summary of the Invention
 - Brief Description of the Drawings (if filed)
 - Detailed Description
 - Claim(s)
 - Abstract of the Disclosure
3. ☒ Drawing(s) (35 U.S.C. 113) [Total Sheets 2]
4. Oath or Declaration [Total Pages 1]
 - a. ☐ Newly executed (original or copy)
 - b. ☐ Copy from a prior application (37 C.F.R. § 1.63(d))
(for continuation/divisional with Box 16 completed)
 - i. ☐ DELETION OF INVENTOR(S)
Signed statement attached deleting inventor(s) named in the prior application, see 37 C.F.R. §§ 1.63(d)(2) and 1.33(b).

* NOTE FOR ITEMS 1 & 13 IN ORDER TO BE ENTITLED TO PAY SMALL ENTITY FEES, A SMALL ENTITY STATEMENT IS REQUIRED (37 C.F.R. § 1.27), EXCEPT IF ONE FILED IN A PRIOR APPLICATION IS RELIED UPON (37 C.F.R. § 1.28).

ADDRESS TO: Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

5. ☐ Microfiche Computer Program (Appendix)
6. Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
 - a. ☐ Computer Readable Copy
 - b. ☐ Paper Copy (identical to computer copy)
 - c. ☐ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

7. ☐ Assignment Papers (cover sheet & document(s))
8. ☐ 37 C.F.R. § 3.73(b) Statement of Power of Attorney
(when there is an assignee)
9. ☐ English Translation Document (if applicable)
10. ☒ Information Disclosure Statement (IDS)/PTO-1449 ☒ Copies of IDS Citations
11. ☐ Preliminary Amendment
12. ☒ Return Receipt Postcard (MPEP 503)
(Should be specifically itemized)
13. ☐ * Small Entity Statement(s) filed in prior application
(PTO/SB/09-12) ☐ Status still proper and desired
14. ☐ Certified Copy of Priority Document(s)
(if foreign priority is claimed)
15. ☐ Other: _____

16. If a CONTINUING APPLICATION, check appropriate box, and supply the requisite information below and in a preliminary amendment:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No: _____ / _____

Prior application information: Examiner _____

Group / Art Unit: _____

For CONTINUATION or DIVISIONAL APPS only: The entire disclosure of the prior application, from which an oath or declaration is supplied under Box 4b, is considered a part of the disclosure of the accompanying continuation or divisional application and is hereby incorporated by reference. The incorporation can only be relied upon when a portion has been inadvertently omitted from the submitted application parts.

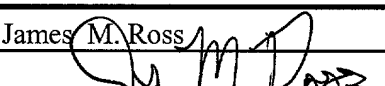
17. CORRESPONDENCE ADDRESS

☐ Customer Number or Bar Code Label

(Insert Customer No. or Attach bar code label here)

or ☒ Correspondence address below

Name	KENYON & KENYON				
Address	Suite 700 1500 K Street, NW				
City	Washington	State	DC	Zip Code	20005-1257
Country		Telephone	(202) 220-4200	Fax	(202) 220-4201

Name (Print/Type)	James M. Ross	Registration No. (Attorney/Agent)	42,115
Signature		Date	3/28/2000

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

**CONFINING AN APPLICATION TO A 32-BIT ADDRESS SPACE SUBSET IN
A 64-BIT PROCESSOR**

FIELD OF THE INVENTION

Embodiments of the present invention relate to a microprocessor. More particularly, embodiments of the present invention relate to confining an application to
5 a 32-bit address space subset in a 64-bit processor.

BACKGROUND OF THE INVENTION

Known 64-bit processors can execute 64-bit applications having 64-bit data types and 64-bit addressing. 32-bit applications typically have 32-bit data types and
10 32-bit addressing. Porting a 32-bit application into a 64-bit environment can result in errors when dealing with 32-bit numbers in a 64-bit environment. For example, a wrong address can be produced when a 32-bit address is incorrectly sign/zero extended. Incorrect content (e.g., an overflow value) can be introduced into the high-order 32 bits of a 64-bit data item when 32-bit data is moved within general purpose registers. An
15 error can also be propagated when an operation (e.g., address generation, etc.) uses an erroneous interim result. In view of the foregoing, it can be appreciated that a substantial need exists for methods and apparatus which can advantageously support execution of 32-bit applications by a 64-bit microprocessor.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is an illustration of an apparatus in accordance with an embodiment of the present invention.

5 FIG. 2. shows a method in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

According to embodiments of the present invention, a processor can include control logic to treat a memory reference of an application as a 32-bit memory
10 reference. In one embodiment, the processor can execute an application, and the application can specify that it uses 32-bit addresses by setting an address space control flag. The processor can determine whether the address space control flag has been set. When the control flag is set, the processor can truncate a generated address into a 32-bit address and either zero-extend or sign-extend to 64-bits the truncated 32-bit address. In
15 one embodiment, a processor can determine whether to zero-extend or sign-extend a truncated 32-bit address based at least in part on whether an address format control flag specifies unsigned addresses (that can be zero-extended) or signed addresses (that can be sign-extended). In another embodiment, a processor can determine whether an address fault control flag has been set. When the address fault control flag is set, in one
20 embodiment, the processor can generate a fault when a generated address is not within a proper 32-bit address space subset. When the address fault control flag is clear, the processor can truncated a generated address and zero/sign-extend the truncated address.

In an embodiment of the present invention, a 32-bit application ported to a 64-bit environment can be executed by a 64-bit processor and operate within a virtual
25 address space. For example, a 32-bit application can operate within an unsigned 0 to 4

GB (gigabyte) virtual address space. Alternatively, a 32-bit application can operate within a signed -2 GB to +2 GB virtual address space.

A 64-bit processor can execute, in an embodiment of the present invention, a 32-bit application that has been ported into a 64-bit environment. The 64-bit processor
5 can include control logic to allow the ported 32-bit application to specify that it uses 32-bit addresses and to instruct the processor to confine the application to a 32-bit address space subset. The processor can confine the application to a 32-bit address space subset by treating a memory reference as a 32-bit access. A memory reference comprising a generated address can be treated as a 32-bit access by truncating the
10 generated address to 32 bits and either zero-extending or sign-extending the truncated generated address to 64 bits.

Confining the application to a 32-bit address subset can compensate for incorrect content that can be introduced into the high order 32 bits of a 64-bit data item when a 32-bit data item is moved and/or part of an operation within a 64-bit general
15 purpose register. One category of such incorrect content relates to wraparound errors. 32-bit operations that wraparound in a 32-bit environment can, when executed in a 64-bit environment, cause the high-order 32 bits of a 64-bit data item to include a value other than 0 ("0" encompasses a value of 0) or 0xFFFF:FFFFH ("FFFF:FFFEH" encompasses a hexadecimal value of FFFF:FFFF). A wraparound error can be caused
20 by, for example, propagation of a carry bit from the low-order 32 bits into the high-order 32 bits. A wraparound error can cause subsequent additional computation errors. Misinterpretation of an operand as a signed or unsigned number can fall under this category. For example, adding 0xFFFF:FFFFH to 0x0000:0002H results in 0x0000:0001H when the 32-bit operands are signed, but results in 0x1:0000:0001H

when the 32-bit operands are unsigned. In a 32-bit environment, the result always wraps to 0x0000:0001H.

Contamination of the high-order 32 bits of a 64-bit number may not be an issue when the computed result is to be stored in a 32-bit data item. When such

5 contamination has occurred and the computed result is to be stored in a 64-bit data item, it can lead to additional errors that affect the results of a computation sequence. For example, an address error can occur. When the computed data item is used to address a data item in memory (e.g., in a 64-bit address space), the computation error can lead to a load or store from an incorrect address.

10 FIG. 1 is an illustration of an apparatus in accordance with an embodiment of the present invention. In an embodiment, a 64-bit processor 100 can execute a 32-bit application that has been ported into a 64-bit environment. The 64-bit processor 100 can include addressing mode control logic 110 having a control flag (e.g., a control bit, etc.) that can specify (e.g., when set, when clear, etc.) that the ported 32-bit application
15 is to be confined to a 32-bit address space subset and use 32-bit addresses. In an embodiment, the processor 100 is running the ported 32-bit application under a 64-bit operating system.

In one embodiment, addressing mode control logic 110 can include an address space control flag 112. In an embodiment, a control flag of addressing mode control
20 logic 110 (e.g., the address space control flag 112) can be set by the ported 32-bit application (e.g., the processor 100 executes an instruction to set the address space control flag 112). In another embodiment, the ported 32-bit application can communicate to a 64-bit operating system that a control flag of addressing mode control logic 110 (e.g., the address space control flag 112) of processor 100 is to be set,
25 and the 64-bit operating system communicates such to processor 100.

When the address space control flag 112 is set, the processor 100 can treat generated address references of the ported 32-bit application as 32-bit addresses by truncating generated address references to 32 bits. For example, memory 120 can store an instruction of ported 32-bit application. Memory 120 can be a cache, an instruction
5 cache, a volatile memory, a non-volatile memory, etc. Instruction execution core 130 can receive the instruction from memory 120. When the instruction is executed, a generated address reference can include 64 bits and be stored in 64-bit register 140. When address space control flag 112 is set, the instruction execution core (or associated logic) can truncate the 64-bit generated address to 32 bits. In an embodiment of the
10 present invention, when a virtual address space used by the ported 32-bit application is a signed address space, the truncated generated address references can be sign extended to 64 bits. When a virtual address space used by the ported 32-bit application is an unsigned address space, the truncated generated address references can be zero extended to 64 bits. In an embodiment, when address space control flag 112 is clear,
15 processor 100 can treat generated address references as 64-bit addresses. In another embodiment, the process can treat all generated address references as 32-bit addresses when an address space control flag 112 is clear and as 64-bit addresses when an address space control flag 112 is set.

In an embodiment of the present invention, addressing mode control logic 110
20 can include address space control flag 112 and an address format control flag 114. Address format control flag 114 can specify whether a 32-bit truncated, generated address reference is to be sign-extended or zero-extended. In an embodiment, when address format control flag 114 is set, processor 100 can treat generated address references as unsigned 32-bit addresses. In such an embodiment, when address format
25 control flag 114 is set, the processor can truncate a 64-bit generated address reference

to 32 bits and zero-extend the truncated, generated address reference to 64 bits. When address control flag 114 is clear, processor 100 can treat generated address reference as signed 32-bit addresses by truncating a 64-bit generated address reference to 32 bits and sign-extending the truncated, generated address reference to 64 bits. In another, embodiment, truncated generated address references are sign-extended when address format control flag 114 is set and zero-extended when address format control flag 114 is clear. In an embodiment, the address format control flag 114 can be ignored unless the address space control flag 112 specifies that the ported 32-bit application is to be confined to the 32-bit address space subset.

10 In a further embodiment of the present invention, addressing mode control logic 110 can include address space control flag 112 and an address fault control flag 116. When address fault control flag 116 is set and address space control flag 112 specifies that a ported 32-bit application running under a 64-bit operating system is to be confined to a 32-bit address space subset, processor 100 can generate an address fault
15 when a generated address reference is not in the proper 32-bit address space subset (e.g., the generated address includes a wraparound error, etc.). For example, after a truncated, generated address reference is sign/zero-extended, the resultant address reference can be compared to the original generated address reference. When the two are different, the address fault can be generated. When the two are not different, the resultant address reference can be used (e.g., as an operand address, as a branch target
20 address, etc.) In another embodiment, when the two are different the address fault can be generated and the resultant address reference can be used. In an embodiment, a resultant address reference can refer to the entire address range covered by an accessed data type. For example, access to a doubleword at address 0xFFFF:FFFEH can include
25 the range 0xFFFF:FFFEH-0x1H.

When address fault control flag 116 is clear and address space control flag 112 specifies that a ported 32-bit application running under a 64-bit operating system is to be confined to a 32-bit address space subset, processor 100 need not generate an address fault when a generated address reference is not in the proper 32-bit address space subset. Rather, the processor 100 can truncate generated address references and sign/zero-extend the truncated, generated address references. In an embodiment, the address fault control flag 116 can be ignored unless the address space control flag 112 specifies that the ported 32-bit application is to be confined to the 32-bit address space subset. In an embodiment, when the address space control flag 112 specifies that the ported 32-bit application is to be confined to the 32-bit address space subset, address faults can be generated when address fault control flag 116 is cleared and address faults need not be generated when address fault control flag 116 is set. Address fault control flag 116, in an embodiment, can be used by a debugger to help detect problems in porting a 32-bit application to a 64-bit environment.

FIG. 2 shows a method in accordance with an embodiment of the present invention. An x-bit application can be executed on a y-bit processor in an x-bit address space subset. In an embodiment, the x-bit application is a 32-bit application and the y-bit processor is a 64-bit processor. In another embodiment, the x-bit application is a 64-bit application and the y-bit processor is a 128-bit processor. In a further embodiment, the x-bit application is a 32-bit application and the y-bit processor is a 96-bit processor. In an embodiment of the present invention, an x-bit application can be executed on a y-bit processor where $y > x$.

An x-bit application can instruct a y-bit processor (processor) to set an address space control flag (box 210), and processor can set address space control flag (box

220). Processor can generate a y-bit address reference as part of execution of an

instruction of x-bit application (box 230). The processor can truncate y-bit address
reference to x bits (box 240). The processor can determine whether the address space
used by x-bit application is signed (decision 250). When the address space used by x-
bit application is signed, the processor can sign-extend x-bit truncated address reference
5 to y bits (box 260). When the address space used by x-bit application is not signed, the
processor can zero-extend x-bit truncated address reference to y bits (box 270). The
processor can determine whether it is to check for an address fault (e.g., by checking
the status of an address fault control flag, etc.) (decision 280). When the processor is to
check for an address fault, the processor can compare the resultant address reference
10 (e.g., the truncated and sign/zero-extended address reference) to the original generated
address reference (e.g., the y-bit address reference generated in box 230) (box 285).
When the resultant address reference does not match the original generated address
reference, the processor can generate an address fault (box 290).

In an embodiment of the present invention, a processor can include addressing
15 mode control logic include an address space control flag, an address format control
flag, and an address fault control flag. The address space control flag can be modified
in any privilege level and can belong to a process context. In an embodiment when a
64-bit processor is running a 32-bit operating system, the address space control flag can
be ignored or considered set. In an embodiment, the address format control flag can be
20 modified in a supervisor privilege level only and can be a global machine flag. The
address format control flag can be set at boot time of a computer system including the
processor. The address fault control flag, in an embodiment, can be modified in
supervisor privilege level only. The address fault control flag can belong to the process
context.

performance is degraded. Software compilers can lessen such degradation of system performance by analyzing the code of the ported application and performing such zero/sign extension operations when necessary. Embodiments of the present invention can compensate for incorrect content without significant performance degradation. A
5 software compiler need not analyze the code to perform necessary zero/sign extensions, and the software need not include additional instructions to fix data values having incorrect content.

Embodiments of methods and apparatus to confine a 32-bit application ported to a 64-bit environment to a 32-bit address space subset have been described. In the
10 foregoing description, for purposes of explanation, numerous specific details are set forth to provide a thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that the present invention may be practiced without these specific details. In other instances, structures and devices are shown in block diagram form. Furthermore, one skilled in the art can readily
15 appreciate that the specific sequences in which methods are presented and performed are illustrative and it is contemplated that the sequences can be varied and still remain within the spirit and scope of the present invention.

In the foregoing detailed description, apparatus and methods in accordance with embodiments of the present invention have been described with reference to specific
20 exemplary embodiments. Accordingly, the present specification and figures are to be regarded as illustrative rather than restrictive.

WHAT IS CLAIMED IS:

- 1 1. A processor comprising:
2 means for executing an instruction of an application of a first bit size ported to a
3 second bit size environment, the second bit size being greater than the first bit size; and
4 means for confining the application to a first bit size address space subset.
- 1 2. The processor of claim 1, wherein the first bit size is 32-bit and the second bit
2 size is 64-bit.
- 1 3. The processor of claim 1, wherein the means for confining includes:
2 means for truncating generated address references of the second bit size to the first
3 bit size; and
4 means for extending to the second bit size the truncated generated address
5 references.
- 1 4. The processor of claim 3, wherein the means for confining includes means for
2 generating an address fault.
- 1 5. The processor of claim 1, wherein the means for extending includes means for
2 determining that the first bit size address space subset is signed address space.
- 1 6. The processor of claim 1, wherein the means for extending includes means for
2 determining that the first bit size address space subset is unsigned address space.

1 7. A processor comprising:
2 a memory to store an instruction of an application ported from a first bit size
3 environment to a second bit size environment, the second bit size being greater than the
4 first bit size; and
5 an instruction execution core coupled to said memory, said instruction execution
6 core to execute the instruction of the application, said instruction execution core to
7 determine that the application is confined a first bit size address subset;
8 generate an address reference of the second bit size as part of execution of the
9 instruction;
10 truncate the generated address reference from the second bit size to the first bit
11 size; and
12 extend the truncated, generated address reference from the first bit size to the
13 second bit size.

1 8. The processor of claim 7, wherein the application ported from a first bit size
2 environment to a second bit size environment is an application ported from a 32-bit
3 environment to a 64-bit environment.

1 9. The processor of claim 7, wherein the instruction execution core is to determine
2 that the application is confined to a first bit size address subset based at least in part on
3 an address space control flag.

1 10. The processor of claim 7, wherein the instruction execution core is to extend the
2 truncated, generated address reference from the first bit size to the second bit size based
3 at least in part on an address format control flag.

1 11. The processor of claim 7, wherein the instruction execution core is to generate
2 an address fault flag based least in part on a comparison of the generated address
3 reference and the extended, truncated, generated address reference.

1 12. The processor of claim 11, wherein the instruction execution core is to generate
2 an address fault flag based least in part on an address fault control flag.

1 13. The processor of claim 7, wherein said memory is a cache memory.

1 14. The processor of claim 7, wherein the processor is a 64-bit processor.

1 15. A method to confine an application to an address space subset, the method
2 comprising:
3 determining that an application is confined a first bit size address subset, the
4 application including an instruction;
5 generating an address reference of the second bit size as part of execution of the
6 instruction;
7 truncating the generated address reference from the second bit size to the first bit
8 size; and
9 extending the truncated, generated address reference from the first bit size to the
10 second bit size.

1 16. The method of claim 15, wherein the application is ported from a first bit size
2 environment to a second bit size environment.

1 17. The method of claim 16, wherein the application ported from a first bit size
2 environment to a second bit size environment is an application ported from a 32-bit
3 environment to a 64-bit environment.

1 18. The method of claim 15, wherein determining that an application is confined a
2 first bit size address subset, the application including an instruction is based at least in
3 part on a address space control flag.

1 19. The method of claim 15, wherein extending the truncated, generated address
2 reference from the first bit size to the second bit size is based at least in part on an
3 address format control flag.

1 20. The method of claim 15, wherein extending the truncated, generated address
2 reference from the first bit size to the second bit size includes sign-extending the
3 truncated, generated address reference from the first bit size to the second bit size based
4 at least in part on an address format control flag.

1 21. The method of claim 15, wherein extending the truncated, generated address
2 reference from the first bit size to the second bit size includes zero-extending the
3 truncated, generated address reference from the first bit size to the second bit size based
4 at least in part on an address format control flag.

1 22. The method of claim 15, wherein extending the truncated, generated address
2 reference from the first bit size to the second bit size includes generating an address
3 fault flag based least in part on a comparison of the generated address reference and the
4 extended, truncated, generated address reference.

- 1 23. The method of claim 22, wherein generating an address fault flag is based least
- 2 in part on an address fault control flag.

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100
1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	

5

DC01-323360

FIG. 1

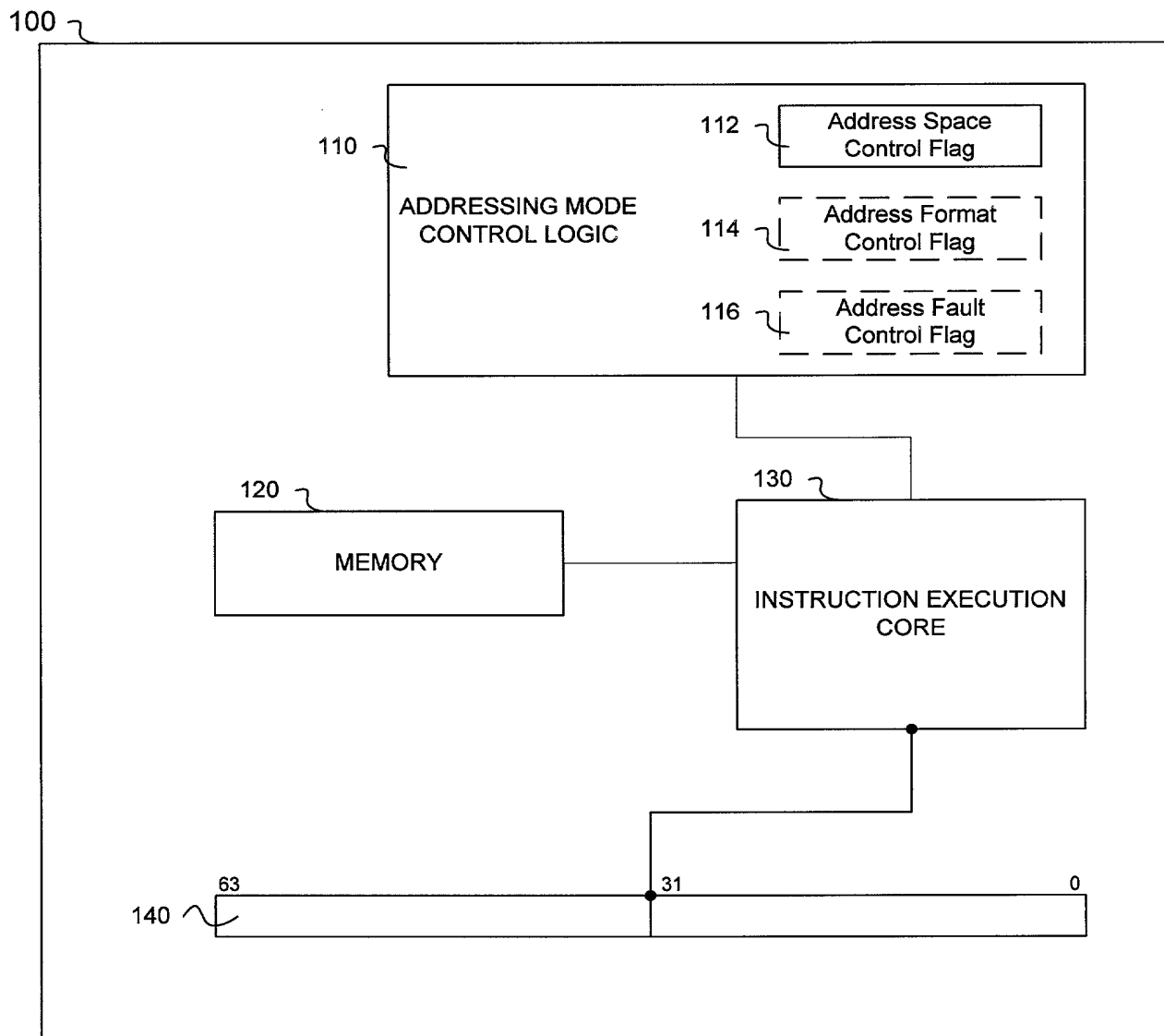


FIG. 2

